Ruby on Rails

Integración Rails con HTTParty

Overview

- Interacción entre Rails y httparty

Agregar al Gemfile

Gemfile

```
31    # bundle exec rake doc:rails generates the API under d
32    gem 'sdoc', require: false
33  end
34
35  # Use ActiveModel has_secure_password
36  # gem 'bcrypt', '~> 3.1.7'
37
38  # Use unicorn as the app server
39  # gem 'unicorn'
40
41  # Use Capistrano for deployment
42  # gem 'capistrano', group: :development
43
44  # Use debugger
45  # gem 'debugger', group: [:development, :test]
46  gem 'httparty', '0.14.0'
47
```

Ejecutar bundle

Cada vez que se ejecuta el bundler es importante reiniciar el servidor.

Mostremos la traducción de un texto

- Mostremos la traducción de un texto
- Generemos el Controller: Data

Crear controller

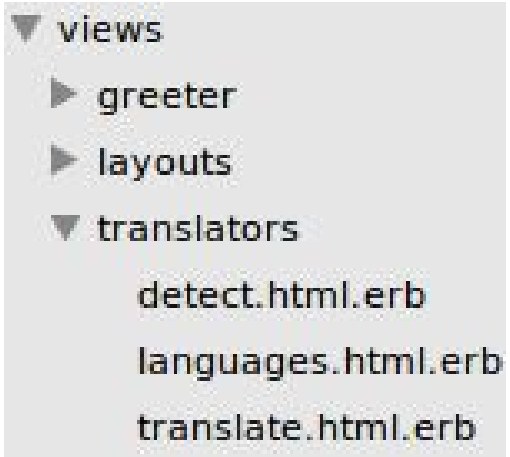
```
rails g controller translators translate detect languages
```

```
MyFirstApp::Application.routes.draw do
  get "translators/translate"
  get "translators/detect"
```

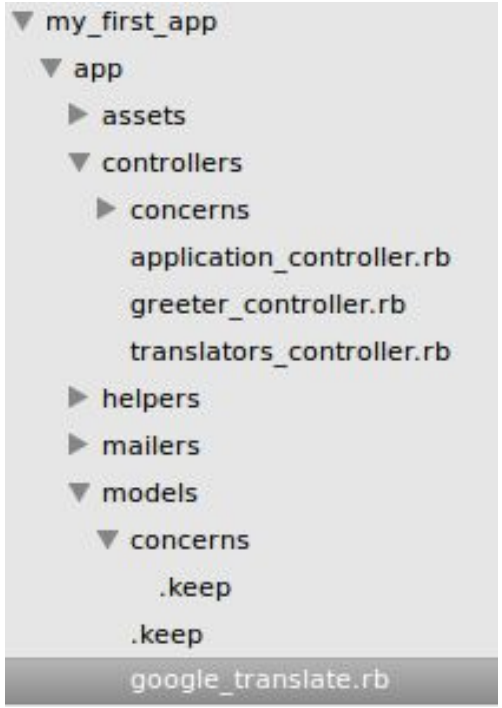
Generated sources

```
MyFirstApp::Application.routes.draw do
  get "translators/translate"
  get "translators/detect"
  get "translators/languages"
```

```
class TranslatorsController < ApplicationController

  def translate
  end

  def detect
  end

  def languages
  end

end
```



Crear un modelo (de acuerdo al servicio)



google_translate.rb × translators_controller.rb ×

```ruby
class GoogleTranslate
  include HTTParty
  default_options.update(verify: false)
  base_uri "https://www.googleapis.com/language/translate/v2"
  format :json

  @@api_key = "AIzaSyBdqiJZunlGxiU5ubdmKlp2VqxqR7GYLsQ"

  default_params key: @@api_key

  attr_accessor :source, :target

  def initialize()
  end

  def initialize(source, target)
    @source = source
    @target = target
  end
```

Controllers -> Plural / Models -> Singular

Implementación del modelo...

```ruby
class GoogleTranslate
  include HTTParty
  default_options.update(verify: false)
  base_uri "https://www.googleapis.com/language/translate/v2"
  format :json

  @@api_key = "AIzaSyBdqiJZunlGxiU5ubdmKlp2VqxqR7GYLsQ"

  default_params key: @@api_key

  attr_accessor :source, :target

  def initialize(source, target)
    @source = source
    @target = target
  end
```

Implementación del modelo

```
def translate(search)
  options = { query: {q: search, source:@source, target:@target}}
  response = self.class.get("", options)
  if response.success?
    response
  else
    raise response.response
  end
end

def self.detect(target_language)
  options = { query: {q: target_language}}
  response = get("/detect", options)
  if response.success?
    response
  else
    raise response.response
  end
end
```

Ejercicio:

Crear un método de clase que invoque al servicio api/**languages** y obtenga una **lista de lenguajes** con los que puede interactuar un determinado **target_language**.

Métodos del controller

```
class TranslatorsController < ApplicationController
  def translate
    @google = GoogleTranslate.new("en", "es")
    @trans = "This is a really good party!"
    @response = @google.translate(@trans)
    @translations = @response['data']['translations'][0]['translatedText']
  end

  def detect
    @trans = "This is a really good party!"
    @detection = GoogleTranslate.detect(@trans)
    puts "#{@detection}"
  end

  def languages
    @target_language = "fr"
    @languages = GoogleTranslate.languages(@target_language)
    puts "Es posible traducir desde los siguientes idiomas: #{@list}"
  end
end
```

Vista: Languages

```
<h1>Detectando idiomas que pueden interactuar con el Francés - <%= @target_language %></h1>

<table border="1">
  <tr>
    <th>Language</th>
    <th>Name</th>
  </tr>
  <% @languages['data']['languages'].each do |lang| %>
    <tr>
      <td><%= lang['name'] %></td>
      <td><%= lang['language'] %></td>
    </tr>
  <% end %>
</table>
```

Vista: Translate

```
<h1>Translated text!</h1>
<p><%= @translations%></p>
```

Vista: Detect

```
<h1>Detected language !</h1>
<h2><%= @detection['data']['detections'][0][0]['language'] %></h2>
```

Entonces...

- Por convención los **controladores** están definidos en **plural**
- Por convención los **modelos** están definidos en **singular**
- Los archivos en el directorio **app/** están automáticamente incluidos.